Project Proposal: Digit Recognizer

Jimmy Lin (xl5224)
Department of Computer Science
University of Texas at Austin
Austin, TX 78712
`jimmylin@utexas.edu`

Jisong Yang (jy7226)
Department of Statistical Scientfic Computation
University of Texas at Austin
Austin, TX 78712
`jsyang993@gmail.com`

1 Introduction

Recognizing hand-written characters has long been a hot topic in artificial intelligence community. Handwriting recognition is the ability of a computer to receive and interpret intelligible handwritten input from sources such as paper documents, photographs, touch-screens and other devices. In the past decades, many probabilistic or non-probabilistic algorithms have been exploited to effectively solve this problem, including K-nearest neighbour, random forest, and well-known neural network.

In this project proposal, we will explain the reason of diving into this problem and demonstrate reader some details of our experimental project. And more importantly, the planned timeline of project progress and our expectation will be present to reader in later section.

2 Motivation

As to the historical application of handwriting recognition, it can be traced back to early 1980s. There are plenty of commercial products incorporating handwriting recognition as a replacement for keyboard input were introduced. The hardware application continued to develop in the following decades. Until the recent years, Tablet PCs can be regarded as a special notebook computer that is outfitted with a digitizer tablet and a stylus, and allows a user to handwrite text on the unit's screen. In addition, highly efficient handwriting recognizers were developed in real world to apply on the zip codes detection.

Plenty of mature solutions has already been developed to existence in solving the handwriting problem. Since 2009, the recurrent neural networks and deep feedforward neural networks developed in the research group of Jrgen Schmidhuber at the Swiss AI Lab IDSIA have outshined many other models in competitions. The fact that existing algorithms are powerful in recognizing digits does not remove possibility of further improvement. The chance of improvement lies in reduction of computational complexity for recognizing digits in higher degree.

3 Project Details

What deserves being mentioned is that the holder of this Kaggle Competition provides us two basic models to better illuminate the problem of digit recognition and the form of given dataset, the MINIST.

3.1 Provided Baseline Models

As mentioned above, implementations of two baseline models are provided by the holder of that kaggle competition. These baselines are respectively K-Nearest Neighbour Model and Random Forest Model. However, the baseline models are extremely computationally expensive since they naively employ the raw data without turning to feature extraction methods.

3.2 Models To Explore

In this project, we should first implement effective feature extraction from the raw image intensity data in MINIST. One possible method for attempt is Principle Component Analysis, which extracts the most distinct components of specified size.

Besides, we will further investigate the performance of a series of machine learning algorithms on digit recognition. Our attempt for this part will involve in neural network algorithm, Support Vector Machine with multiple class labeling, and treat Deep Learning algorithm as extension. For specific implementation of other algorithms, we will refer to existing libraries.

3.3 Timeline

The project timeline is shown as follows:

- **March. 23 - April. 02**. Propose project. Run two provided baseline models and evaluate its performance. Implement feature extraction PCA.
- **April. 02 - April. 12**. Explore performance of new models in given dataset. Tasks may involves in reading relevant paper. Current plan at this stage is to achieve neural network algorithm and support vector machine on the preprocessed data.
- **April. 13 - April. 18**. Draft initial version of project report . Record and compare temporal and spatial perfermance of all attempted algorithms.
- **April. 19 - April. 25**. Extension: Further explore some more advanced models, say, Deep neural network. (optional)
- **April. 25 - May. 2**. Put down new progress at second-time exploration and proofread the project report.

The detailed task assignments are shown as follows:

Jisong Yang: modify project proposal, PCA, Neural Network, DNN(optional), final project report
Jimmy Lin: draft project proposal, SVM, DNN(optional), final project report

4 Conclusion

The main objective of this project is to apply various existing machine learning algorithms on the long-lasting digit recognition problem and evaluate the performance of each algorithm on solving this particular problem. In addition, we leave another problem, the feature extraction, as our secondary focus.

Our wish is to achieve a model with digit recognition accuracy as close to 100% as possible. Actually, about 10 teams have already fulfill perfect recognizer, identifying given digit image with no error! But for us, a accuracy over 90% is acceptable for this experimental project.

5 References

[1] Liu, C. L., Nakashima, K., Sako, H., & Fujisawa, H. (2003). Handwritten digit recognition: benchmarking of state-of-the-art techniques. *Pattern Recognition*, **36(10)**, 2271-2285.

[2] Le Cun, B. B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1990). Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems.*

[3] LeCun, Y., Jackel, L. D., Bottou, L., Brunot, A., Cortes, C., Denker, J. S., & Vapnik, V. (1995, October). Comparison of learning algorithms for handwritten digit recognition. In *International conference on artificial neural networks* **(Vol. 60)**.

[4] Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, **18(7)**, 1527-1554.

[5] Ciresan, D. C., Meier, U., Gambardella, L. M., & Schmidhuber, J. (2010). Deep, big, simple neural nets for handwritten digit recognition. *Neural computation*, *22(12)*, 3207-3220.